# 쿼드 스트립을 이용한 흐름의 시각화에 관한 연구*

김기섭, 이원형
중앙대학교 첨단영상대학원

# Flow Visualization using quad-strip Mapping Method

**Ki-Seob Kim**, Won-Hyung Lee
The Graduate School of Advanced Imaging Science, Multimedia and Film
Chung-Ang University

**요 약**

Cabral과 Leedom이 처음 제시한 LIC(Line Integral Convolution)은 기존의 DDA 알고리즘에 기반한 회선 처리방식보다 우수한 것으로 인정받았고 과학적 시각화 분야에 많은 공헌을 하였다. 그리하여 이것을 기반으로 하는 많은 논문들이 나왔다. 우리는 본 논문에서 기존 LIC의 선 중심의 스트림 라인이 아니라 openGL의 기하 프리미티브인 쿼드 스트립을 이용한 어느 정도의 두께를 가진 스트림 라인을 생성하여 보다 새롭고 다양한 흐름의 시각화 기술을 제시하고자 한다.

## 1. 서론

방향정보를 가지고 있는 벡터 필드를 이용하는 다양한 연구 분야가 있다. 그 중의 하나인 과학적 시각화 분야는 원래 유체흐름과 같은 기상관측이나 의학 같은 분야에 집중을 하였으나, 요즈음에는 컴퓨터 그래픽스에서 주목받고 있는 비사실적 렌더링(Non-Photorealistic Rendering) 분야의 하나로 각광받고 있다. Scientific Visualization 분야는 Flow Visualization 분야의 두 흐름, 즉 Van Wijk의 Spot Noise 알고리즘을 기반으로 하는 Flow Visualization과 Cabral과 Leedom이 제안한 Line Integral Convolution이 있다. Cabral과 Leedom이 제안한 Line Integral Convolution(LIC)는 그 동안의 DDA에 기반한 회선처리의 단점인 균형에 민감하다는 것과 정확성의 결여를 극복하고 새로운 흐름의 시각화 방법으로 주목받았다. 그러나 이들의 방법도 처리시간이 오래 걸린다는 단점이 있어, 이 후의 논문들은 Cabral과 Leedom이 발표한 LIC를 보완하는 논문들이 주류를 이루게 되었다. 본 논문에서 제안하는 흐름의 시각화 기술은 LIC에서 제시하는 방법인 스트림라인(streamline) 기반인 벡터 필드를 사용하지만 기존의 방법인 단순한 선(line) 기반의 스트림 라인이 아니라 어느 정도의 두께를 가진 스트림 라인을 생성하고 생성한 스트림 라인을 벡터필드에 맵핑을 하는 방법을 사용하여 기존의 벡터 필드 구성과는 다른 방법을 제시하였다.

* 본 연구는 한국학술진흥재단 두뇌한국21사업의 지원을 받아 수행되었습니다.

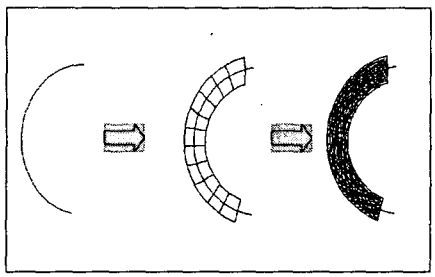
## 2. 관련 연구

밀도가 높은 벡터 필드의 시각화를 위한 가장 일반적인 방법은 텍스쳐에 기반한 방법이다. Van Wijk[2]에 의해 제안된 spot noise 방법은 벡터필드에 평행하는 직선을 따라 회선을 시켜주는 것이다. 또 다른 방법은 1993년 Cabral과 Leedom[1]에 의해서 처음으로 제시된 LIC이다. 이 방법은 Input Image로는 화이트 노이즈 이미지와 벡터필드를 입력받아 벡터필드에서 각각의 좌표마다 스트림라인을 생성하고 각각의 스트림 라인마다 회선처리를 하여 결과 이미지를 얻는다. 이 기술은 입력영상과 출력영상은 동일한 차원(dimension)의 영상이 된다. 즉 입력영상이 2차원 영

상이면 출력영상도 그러하고 입력영상이 3차원 영상이면 출력영상도 3차원 영상이 생성된다. 이 LIC기술이 발표된 후, 이 기술을 추종하는 많은 논문들이 발표되었는데 그 방향은 기존의 LIC가 처리시간인 오래 걸린다는 단점을 극복하려는 방향과 다른 방향은 LIC를 이용하여 어떠한 새로운 응용분야를 탐색하려는 방향으로 나누어 볼 수 있다. Stalling과 Hege[3]는 LIC 기술을 발전 시키기 위해 스트림 라인을 따라 일치와 불일치 검색을 최적화하는 방법을 제안하였다. Stalling과 Hege는 스트림 라인 회선시 Cubic Hermite-Interpolation을 사용하고 선택적으로 영상대조를 증가시키기 위해 Direction gradient 필터를 사용한다. 그리고 흔히 Stalling과 Hege의 방법을 빠른 LIC(Fast LIC)라고 부른다. Forssel[4]은 LIC 알고리즘을 곡선형의 격자에 적용시키는 방법을 제안하였고 Okada와 Kao[5]는 이미지 대조를 증가시키고 흐름의 특징을 강조하기위해서 후 필터처리기술을 사용하였다.

## 3. 쿼드스트립을 이용한 흐름의 시각화

본 논문에서 제안하는 스트림 라인 생성과정을 살펴보면 아래의 그림과 같다.



[ 그림 1 ]

기존의 LIC 방법에서는 단순한 벡터 필드상에서 각 픽셀마다의 벡터에서 그 벡터들을 하나의 선으로 연결하여 스트림 라인을 구현하였다. 이러한 방법은 모든 픽셀에서 스트림 라인을 생성해야 하였기 때문에 많은 처리시간이 소모되는 단점이 있었다. 이것을 개선하고자 본 논문에서는 스트림 라인을 단순한 선의 개념에서, 어느 정도의 두께를 가지는 스트림 라인을 만들고 그것을 텍스쳐 맵핑을 시키는 방법을 제시하였다. 각 픽셀상에 너무 많은 스트림 라인들이 생성이 되면 그 값들이 그레이 스케일 범위 안에 놓여지는 경향이 있어서 결과 이미지의 대조를 상실하게 되는 어려움이 있다. 그것을 방지하기 위해서 OpenGL 의 블렌딩 함수인 glBlendFunc()을 이용하였다. 스텐실 버퍼를 OpenGL에 사용하기 위해서 스텐실 테스트를 활성화시킨 다음 스텐실 테스트의 비교함수와 작용방식을 설정하였다. 스텐실 비교함수를 설정할 때 쓰이는 glStencilFunc()를 위의 알고리즘과 같이 설정하였는데 제안하는 방법은 모든 픽셀에서 스텐실 테스트를 통과하는 것 이므로 GL_ALWAYS와 같이 설정하였다.

```
● glBlendFunc(gl_SRC_ALPHA,
        gl_ONE_MINUS_SRC_ALPHA);

● glStencilFunc(GL_ALWAYS, 0, 0);
    glStencilOp(Glenum fail, Glenum zfail, Glenum zpass);
```

다음으로 스텐실 작용 방식을 설정해야 하는데, 이 때 사용하는 함수는 glStencilOp()이다. 위의 알고리즘에서 보는 바와 같이 fail 인자는 스텐실 테스트를 통과하지 못했을 때 적용될 작용을 의미하며, zfail은 스텐실 테스트는 통과했으나 깊이 테스트는 통과하지 못했을 때 적용될 작용을 의미한다. 그리고 zpass는 스텐실 테스트와 깊이 테스트를 모두 통과했을 때 적용될 작용을 의미한다. 본 논문에서는 스텐실 버퍼를 항상 통과하는 것이 목적이고 모든 픽셀에서 스텐실 버퍼의 값을 증가시켜야 하므로 'GL_INCR'로 설정하였다

입력 벡터 필드에서의 점 $p$ 의 좌표에서 매개곡선 $P(p,s)$는 벡터 필드에서 전진방향과 그 반대방향으로 어느 정도의 거리 $L$를 두면서 각각의 좌표들을 통과하면서 생성된다. 출력 픽셀 $F'(p)$는 스트림 라인을 따라 계산된 입력 이미지 $F$ 값들의 가중치 합이다. 수식 ①이 그것을 나타내고 있다.

$$F'(p) = \frac{\int_L^t F(P(p,s))k(s)ds}{\int_L^t k(s)ds} \qquad ①$$

이것을 이산 형식으로 나나내면 아래 수식② 와 같다

$$F'(p)=\frac{\sum_{i=0}^{l}F(P_i)h_i}{\sum_{i=0}^{l}h_i} \qquad ②$$

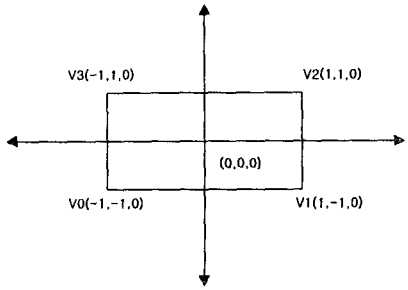
LIC_Convolve(**pixels** F, **vectorField** V, **pixels** F')
**For each** p vector field position in V
$F'(p) = 0$
**For each** $Pi$, defined over the local vector streamline at p
$F'(P) = F'(p)+f(Pi)hi$
$F'(p) = F'(p)/\Sigma hi$

그리고 스트림 라인을 따라서 전진방향과 그 반대 방향이 있으므로 수식③으로 나타낼 수 있다. 수식③은 출력 픽셀의 완전한 수식을 나타낸다.

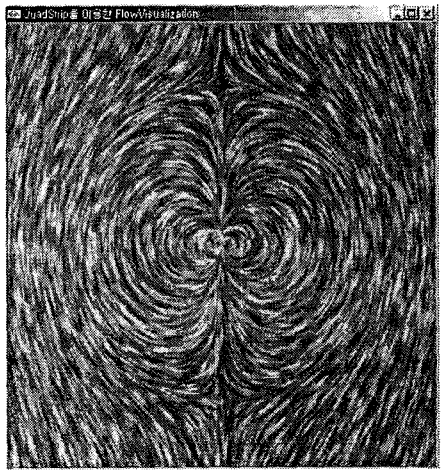
$$F'(p)=\frac{\sum_{i=0}^{l}F(P_i)h_i+\sum_{i=0}^{l'}F(P'_i)h'_i}{\sum_{i=0}^{l}h_i+\sum_{i=0}^{l'}h'_i} \qquad ③$$

## 4. 구현 및 결과

벡터필드의 구성을 위해서 초기의 쿼드 좌표를 아래와 같이 설정하였다. 원점을 중심으로 하여 점 4개를 찍어서 쿼드를 표시하였다




```
V0[X] = -1.f; V0[Y] = -1.f; V0[Z] = 0.f;
V1[X] = 1.f; V1[Y] = -1.f; V1[Z] = 0.f;
V2[X] = 1.f; V2[Y] = 1.f; V2[Z] = 0.f;
V3[X] = -1.f; V3[Y] = 1.f; V3[Z] = 0.f;
```



[ 그림 2 ]

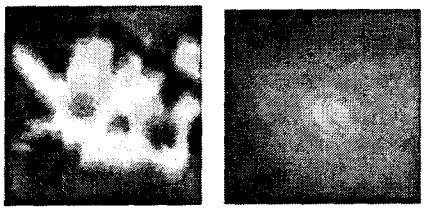
위의 [그림 2]은 입력 영상으로는 화이트 노이즈를 사용하여 만든 결과 이미지이다.

또한 [그림 3]는 본 논문에서 제안한 방법을 이미지 제작 툴인 포토샵(photoshop)과 비교한 그림이다. [그림3]의 (b)(c)는 포토샵을 이용한 이미지이고 아래줄의 [그림3]의 (b)'(c)'는 본 논문에서 제안한 방법을 사용하여서 만든 이미지이다.

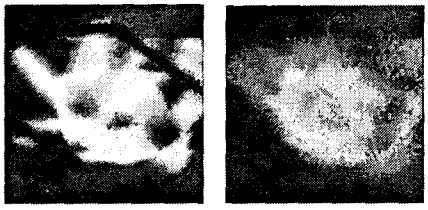
[그림 3]의 (b)(b)'는 가우시안 블러링 필터 길이를 5로 적용한 결과 이미지이고 (c)(c)'는 가우시안 블러링 필터길이 20을 적용한 이미지이다. 포토 샵을 이용한 단순한 가우시안 필터를 적용하면 [그림 3]의 (b)(c)에서 보는 바와 같이 블러링 효과만이 적용되어 흐릿한 효과만이 생긴다. 그리하여 [그림 3]의 (c)처럼 필터길이 20과 같이 그 길이를 길게 하였을 경우는 어떤 이미지인지 알아보기가 힘들다.

그러나 본 논문에서 제안한 방법을 이용하여 가우시안 블러링을 적용한 이미지들은 [그림 3]의 (b)'(c)'에서 보는 바와 같이 포토 샵에서 실험한 이미지들과는 달리 이미지의 퀄리티가 생생하고 환상

(원본이미지)



(b) (c)



(b') (c')

[ 그림 3 ]

적인 효과마저 느끼게 한다. 마치, 뿌연 유리창 밖에 피어있는 노란 개나리꽃을 보는 것과 같은 느낌을 들게 한다.

## [참고문헌]


[1] B. Cabral and C. Leedom. Imaging vector fields using line integral convolution. In proceedings of SIGGRAPH 93, pp.263-270. ACM SIGGRAPH 1993

[2] J. van Wijk. Spot noise; Texture synthesis for data visualization. Computer Graphics, 25(4):309-318, 1991

[3] D. Stalling and H.C. Hege. Fast and resolution independent line integral convolution. In Proceedings of SIGGRAPH '95, pp 249-256. ACM SIGGRAPH 1995

[4] L.K. Forssell and S. D. Cohen. Using line integral convolution for flow visualization; Curvilinear grids, variable-speed animation, and unsteady flows. IEEE Transactions on Visualizatin and Computer Graphics. 1(2); 133-141, 19995

[5] A. Okada and D. L. Kao. Enhanced line integral convolution with flow feature detection. In Proceedings of IS&T/SPIE Electronic Imaging '97, pp.206-217

[6] Han Wei Shen and D. L. Kao. A new line integral convolution algorithm for visualizing time-varying flow fields. IEEE Transactions on Visualization and Computer Graphics, 4(2), 1998

[7] V. Verma, D. Kao, A. Pang. Plic: Bridging the gap between streamlines and lic. In Proceedings of Visualization '99, pp 341-348. IEEE Computer Society Press 1999.

[8] B. Jobrard and W. Lefer. Unsteady flow visualization by animating evenly-spaced streamlines. Computer Graphics Forum(Proceedings of Eurographics 2000), 19(3), 2000

[9] M. H. Kiu and D. Banks. Multi-frequency noise for LIC. In Proceedings of Visualization '96. pp 121-126. IEEE Computer Society Press, 1996

[10] G. Truk and D. Banks. Image-guided streamline placement. In Proceedings of SIGGRAPH'96. pp.453-460. ACM SIGGRAPH, 1996.

[11] Han-Wei Shen, Using line integral convolution to visualize dense vector fields. Computer In Physics, VOL 11, NO 5, SEP/OCT pp.474-478, 1997